Journal of Biomedical Informatics 56 (2015) 395–405

Contents lists available at ScienceDirect

Journal of Biomedical Informatics

journal homepage: www.elsevier.com/locate/yjbin

Automated generation of directed graphs from vascular segmentations ☆

Brian E. Chapman [a,*], Holly P. Berty [b], Stuart L. Schulthies [c]

[a] University of Utah, Department of Radiology, 729 Arapeen Drive, Salt Lake City, UT 84108, United States
[b] University of Pittsburgh, Department of Biomedical Informatics, 5607 Baum Boulevard BAUM 423, Pittsburgh, PA 15206-3701, United States
[c] University of Utah, Department of Mathematics, 155 S 1400 E, Salt Lake City, UT 84112-0090, United States

ARTICLE INFO

Article history:
Received 19 February 2015
Revised 1 July 2015
Accepted 2 July 2015
Available online 9 July 2015

Keywords:
Medical imaging
Segmentation
Feature recognition

ABSTRACT

Automated feature extraction from medical images is an important task in imaging informatics. We describe a graph-based technique for automatically identifying vascular substructures within a vascular tree segmentation. We illustrate our technique using vascular segmentations from computed tomography pulmonary angiography images. The segmentations were acquired in a semi-automated fashion using existing segmentation tools. A 3D parallel thinning algorithm was used to generate the vascular skeleton and then graph-based techniques were used to transform the skeleton to a directed graph with bifurcations and endpoints as nodes in the graph. Machine-learning classifiers were used to automatically prune false vascular structures from the directed graph. Semantic labeling of portions of the graph with pulmonary anatomy (pulmonary trunk and left and right pulmonary arteries) was achieved with high accuracy (percent correct $\geqslant 0.97$). Least-squares cubic splines of the centerline paths between nodes were computed and were used to extract morphological features of the vascular tree. The graphs were used to automatically obtain diameter measurements that had high correlation ($r \geqslant 0.77$) with manual measurements made from the same arteries.

1. Introduction

An important imaging informatics task is to help medical imaging evolve from a primarily qualitative to a primarily quantitative discipline. One aspect of this is extracting quantitative and computable features from the image. Being able to do this in a (nearly) automated method would allow prospective collection of quantitative features with minimal impact on current workflow and the retrospective processing of large numbers of cases archived in institutional PACS. While some quantitative feature extraction can be done directly on the original image, typically, extraction involves identifying subregions of the image that constitute particular objects of interest within the image. Sub-images may be geometric subunits of the image or collections of connected voxels that represent an object or feature of interest. A segmentation of a medical image is a binary labeling of the pixels (2D) or voxels (3D) that constitute the image, where each voxel that is part of the object of interest is given one label (e.g. 1) and all other voxels are given another label (e.g. 0).

After segmentation, the labeled voxels are simply an unordered list, and, depending on the complexity of the segmented object, may need to be ordered into substructures in order to facilitate processing or reasoning. In this paper we present a process for ordering a vascular skeleton into the constituent parts of the vascular tree so that quantitative, computable features can be extracted from the original medical images. Our method uses graph-based techniques to recognize critical features within the skeleton (bifurcations, endpoints, and centerlines). Once the skeletal tree structure is recognized, each voxel within the segmentation is mapped to the appropriate graph edge, facilitating characterization of morphological features of specific vascular segments. For this paper we focus on 3D (volumetric) vascular images and how to structure the original unordered list of prior segmented voxels so that vascular-specific features, such as bifurcation angles or segment diameters, can be automatically extracted. The basis for this structuring is extracting the skeleton of the vascular tree.

The vascular skeleton can be extracted directly from the original (gray scale) image based on the curvature properties of the image, using, for example, ridge traversal [1]. However, these techniques are computationally expensive and may not be ideal for extracting the underlying structures in all cases, since the skeletal extraction is explicitly connected to global models that might be difficult to

☆ This work was supported in part by NIH Grants NHLBI R01 HL087119, U54HL108460, by the Department of Biomedical Informatics at the University of Pittsburgh, and by the Office of the Senior Vice President for Health Sciences at the University of Utah.

* Corresponding author.
E-mail address: brian.chapman@utah.edu (B.E. Chapman).

http://dx.doi.org/10.1016/j.jbi.2015.07.002
1532-0464/

match in the presence of confounding anatomical structures. By extracting the skeleton from a segmented image, a wide variety of segmentation techniques can be used to accurately capture the vascular structures of interest. Given a segmented image, researchers have proposed a variety of means of extracting the skeleton, using, for example, wave propagation [2] or tracing optimal paths using Dijkstra's algorithm (e.g. [3]). However, these methods are sensitive to the cost functions selected for the algorithm and the shortest path through a curve is not at the center of a vessel. Alternatively, parallel thinning techniques are model-free, morphology-based approaches to skeleton extraction [4].

However they are created, automatically generated skeletons will almost inevitably require pruning of spurious centerlines. This pruning may be based on simple features such as centerline length [5] or by trying to recognize non-physiological branching angles [6]. Consequently, we explore using machine learning techniques to automatically prune spurious centerlines.

Given a skeleton, the task remains to recognize endpoints and bifurcations in order to define the underlying vascular structures.

The ability to automatically extract the pulmonary arterial structure has a variety of important implications. First, it could help in the development of computer-aided diagnosis algorithms for pulmonary vascular diseases. For example, these automated techniques could be used to quantify vascular geometry as depicted in volumetric medical images (CT or MR), to assist in the diagnosis of pulmonary arterial hypertension (PAH). While manual arterial measurements have been shown to differentiate PAH subjects from normal subjects [7–10], automated feature extraction for PAH diagnosis would aid radiology workflow. Further, automated measurements would allow a more comprehensive disease characterization based on a fuller assessment of the arterial tree, rather than being limited to a few arteries. Similarly, automated vascular tree extraction could help in the design of computer-aided diagnosis of pulmonary embolism by eliminating non-arterial structures prior to the search for filling defects or by limiting the search to a vascular depth that is deemed clinically significant (e.g. excluding sub segmental arteries). These automated techniques could also be used for large-scale, retrospective image-based analysis for quality assurance purposes or for image-based phenotyping for knowledge discovery in conjunction with additional clinical and genomic information.

We used the Python programming language for our tool development, incorporating unmodified third-party, open source image analysis and visualization libraries, such as the Insight Toolkit [11] and the Visualization Toolkit [12] that were accessed through Python wrappers. These tools and all dependencies are easily installed on multiple platforms. We evaluated our methods on a set of 116 CT pulmonary angiography (CTPA) images.

2. Materials and methods

We begin with a description of our data collection and vascular segmentation followed by the vascular graph generation process, where we detail how we map the voxels from the original segmentation to the graph edges to build a complete representation of the vascular structure. Finally, we describe our machine learning approach for pruning the spurious segments from the graph and thus improving the semantic labeling of our models.

2.1. Data collection

For this study, we used a set of 116 de-identified CTPA exams that had been collected for other studies. All images were acquired at the same institution with diagnostic imaging settings using similar multi-row detector helical CT scanners reconstructed with slice thickness ranging from 0.625 mm to 5.0 mm; the modal thickness was 1.25 mm.

2.2. Vascular segmentation and skeleton generation

Automated segmentation of medical images remains one of the most difficult problems in medical image processing [13]. While machine learning techniques for image segmentation have had a great impact on segmentation of traditional 2D scenes, a similar impact has not been seen in medical imaging where the number of available cases is much smaller and the cost of annotating images for training much higher [14]. Consequently, several researchers have introduced unsupervised learning techniques [14,15]. Nonetheless, within the sub-domain of vascular segmentation, the state-of-the art techniques still rely on rules applied to tubular models of vascular structures, which are very good at extracting the peripheral pulmonary arteries but generally cannot capture the central arteries (pulmonary trunk and left and right pulmonary). These central vessels, which are presumably the most informative for diseases such as pulmonary arterial hypertension, are difficult to segment automatically because of their proximity to confounding structures such as the heart and the aorta. Since our primary interest is in structuring a vascular segmentation rather than in developing novel segmentation algorithms, we used the geometric level set algorithm in ITK-SNAP [16] to generate an initial segmentation of the vasculature followed by hand editing of the resulting segmentation using the paintbrush tool in ITK-SNAP. We felt this would produce segmentations similar to a quasi ideal automated technique. Since this work was motivated in part by the problem of automated characterization of pulmonary hypertension, we focused our segmentations on the central pulmonary arteries (pulmonary trunk and left and right pulmonary arteries).

The level-set segmentation required the user to both provide seed points from where to start the segmentation and either an intensity or gradient mapping that drives the evolution of the segmentation. We chose to use intensity maps because our initial experience was that the intensity maps generally produced less leakage of the segmentation into non-vascular structures. Seed points were placed in the pulmonary trunk and the left and right pulmonary arteries (Fig. 1). The segmentation was allowed to proceed until the pulmonary trunk and left and right pulmonary arteries were fully captured. The amount the segmentation bled into non-vascular structures and how far down the vascular tree the segmentation proceeded varied depending on the characteristics of the particular CTPA exam.

Manual editing of the vascular segmentation was done by reviewing the segmentation on a slice-by-slice basis. Using the paintbrush tool in ITK-SNAP, any observed leakages of the segmentation into non-vascular structures were deleted (Fig. 2). We did not, however, delete any vascular structures beyond the central arteries that were included in the segmentation. Consequently the complexity of the pulmonary arterial tree that was captured varied on a case-by-case basis.

2.2.1. Segmentation preprocessing

We observed that imperfections in the segmentation, such as small holes and surface irregularities, could lead to great difficulty in cleanly generating the vascular graph model skeleton. Consequently, some preprocessing of the segmentation had to be performed prior to generating the skeleton and then the graph. We explored using common binary filters to reduce these imperfections prior to 3D parallel thinning. Specifically, we explored using median filtering [17] and morphological closing [18], alone and in combination. Both filters were implemented using ITK filters (itkBinaryMedianImageFilter and itkBinaryMorphologicalClosingImageFilter respectively) with either $(1,1,1)$ or $(2,2,2)$ kernels.

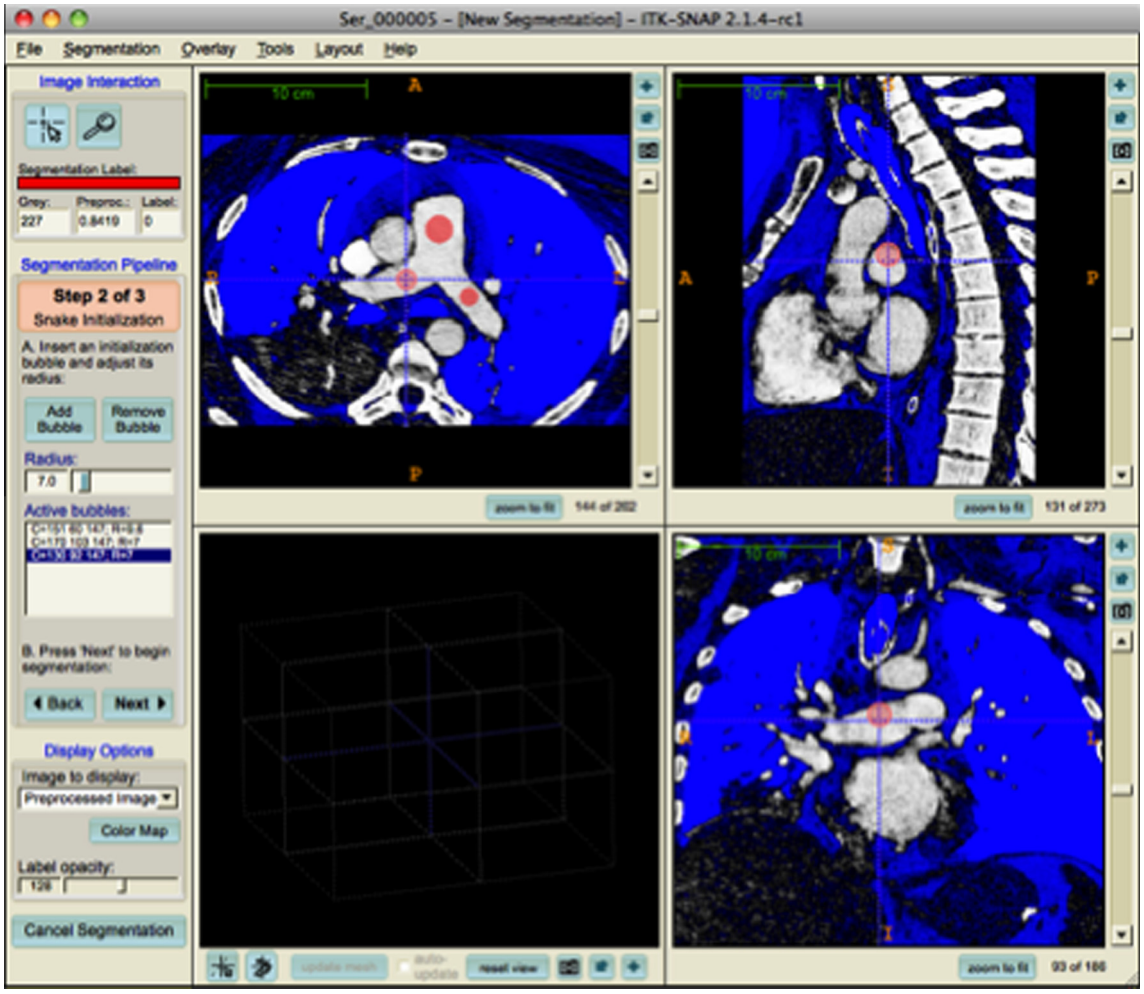


Fig. 1. Screenshot of the segmentation setup using ITK-SNAP. The intensity mapping is overlaid on the original grayscale image (blue to white mapping) and the three segmentation seed bubbles (red) are placed in the pulmonary trunk and left and right pulmonary arteries. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

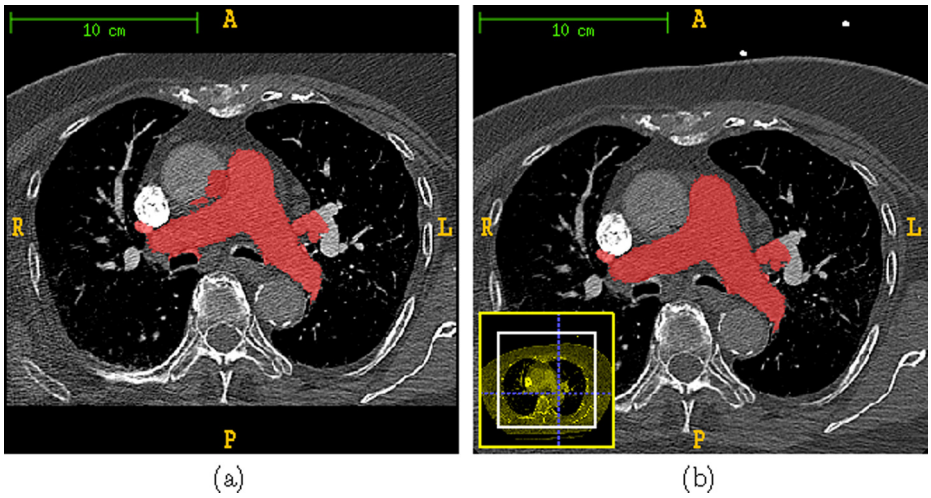


Fig. 2. Editing of segmentations. (a) Original segmentation demonstrating bleeding into the aorta. (b) Segmentation after editing to eliminate major leakages.

We used the heuristic of minimizing the number of edges in the undirected vascular graph (see Section 2.3.1) to assess preprocessing performance. Although this minimization does not make sense in the extreme (a zero edge graph would be optimal), we found in practice this was a reasonable measure, since the filtering reduced spurious edges and not edges corresponding to true anatomical

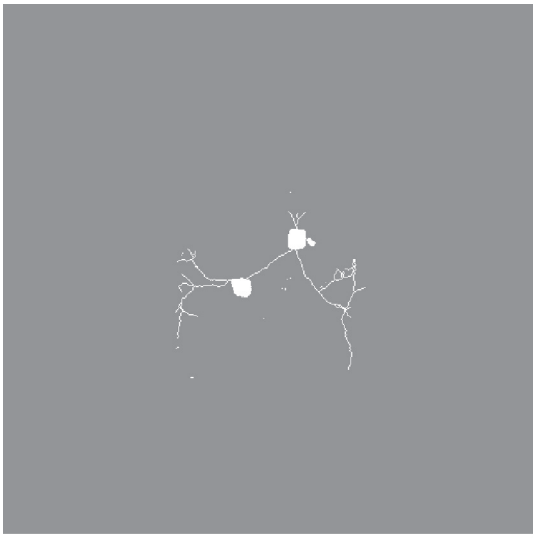
structures. An illustration of the value of this preprocessing is shown in Fig. 3.

2.2.2. Skeleton generation

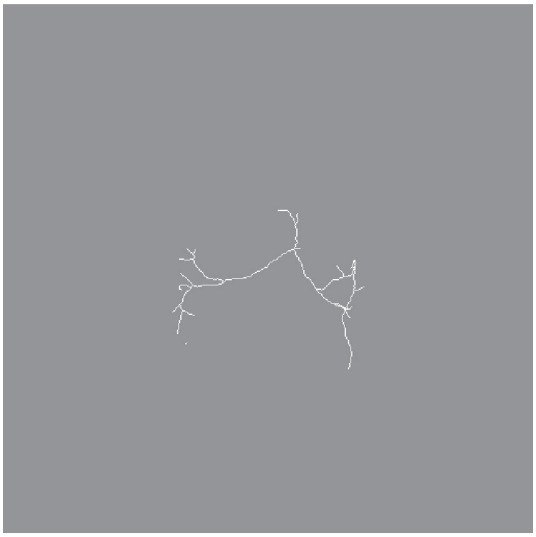
We generated the skeleton of the binary image using the decision-tree based parallel thinning algorithm of Homann [19]. The C++ code was downloaded from *The Insight Journal* website and was compiled unmodified against an ITK version 3.x library. An example segmentation and the corresponding skeleton are shown in Fig. 4.

2.3. Graph generation

Given the skeleton of the vascular tree segmentation, we generated the graph representations of the vasculature. First, we translated the skeleton image to an undirected graph of the vascular tree. Second, we created a directed graph representation of the vascular tree. Finally, we used machine learning techniques to prune spurious nodes and edges from the graph based on features obtained directly from the graph and from features obtained by mapping the full 3D segmentation voxels to the graph. These steps are described below.

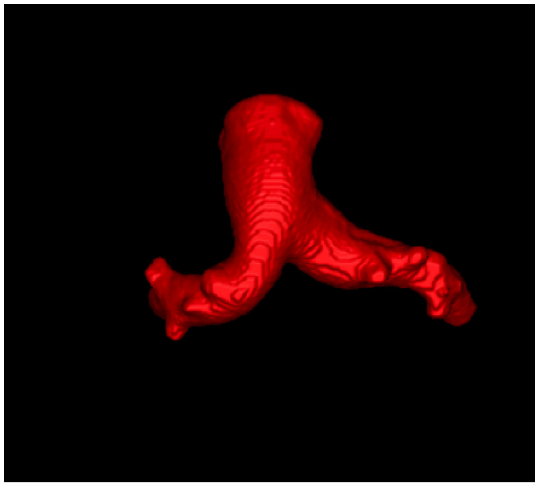


(a) Skeleton generated from unprocessed ITK-SNAP segmentation.

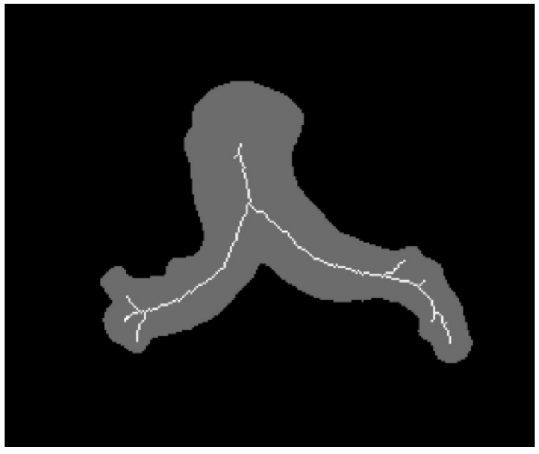


(b) Skeleton generated after processing segmentation with median and morphological closing filters.

Fig. 3. Example skeleton generation before and after preprocessing to reduce surface irregularities.



(a) Surface rendering of ITK-SNAP segmentation.



(b) MIP image of segmentation with skeleton superimposed.

Fig. 4. Example segmentation and skeleton generated with 3D parallel thinning.

2.3.1. Undirected graph representation

Given a skeletal image (I_{skel}) generated from a binary segmented image (I_{seg}) by Homann's method, let S_{skel} denote the set of nonzero voxels v in the image. To recognize vascular structures, we first mapped S_{skel} into structures that correspond to the underlying anatomy. We achieved this using a multi-step process based on graphs using the NetworkX [20] Python library.

First, we created an undirected graph (G_u) where each $v \in S_{skel}$ was added as a node in the graph. Each node (n) had as attributes the (i,j,k) location in the image matrix and the (x,y,z) location in the world coordinate system defined for the image. Edges were added between any nodes coexisting within a $3 \times 3 \times 3$ neighborhood in I_{skel}. This algorithm is shown in Fig. 5.

Once G_u was generated, the degree of each node was used to determine which of three types of voxel the node represented: (1) degree-one nodes corresponded to endpoint voxels (n_E), (2) degree-two nodes corresponded to centerline voxels (n_C), and (3) degree-three (or greater) nodes corresponded to bifurcation voxels

```
input  : S_skel, I_skel
output: G_u
foreach v ∈ S_skel do
    neighbors = FindNeighbors (I_skel, v);
    foreach v' ∈ neighbors do
        AddEdge(G_u, v, v') ;
    end
end
```

Fig. 5. Algorithm for generating undirected graph G_u from a skeletal image I_{skel}.

(n_B); anatomically we would only expect bifurcation nodes, but errors in the segmentation process may result in nodes with more than three neighbors. An example undirected graph is shown in Fig. 6.

2.3.2. Directed graph representation

A directed graph (G_d) was next generated using the set of all endpoint nodes S_{n_E} and the set of all bifurcation nodes S_{n_B} identified in G_u. Since a root node (n_r) is required for a directed graph, we developed a simple heuristic algorithm for identifying the node corresponding to the termination of the pulmonary trunk (n_{PT}) in the segmentation. We identified n_{PT} using image orientation information extracted from the headers of the original medical images. Let x_m be the medial location of the image, y_p be the most posterior location of the image, and z_s be the most superior location of the image and $v_{mps} = (x_m, y_p, z_s)$ be the voxel at that location. n_{PT} was then the degree-one node that minimized

$$n_{PT} = \underset{n_e \in S_e}{\text{minimize}} \|n_e - v_{mps}\| \quad (1)$$

n_r was then equated with n_{PT}. The accuracy of proper identification of n_r was assessed by visual review of 2D projections of the graphs.

A bidirectional Dijkstra algorithm was used to find the shortest path ($P_{(n_e,n_r)}$) between each n_e and n_r in G_u. $P_{(n_e,n_r)}$ was then split into segments at each bifurcation node included in the path. The endpoints of each segment were either degree-one nodes (vascular endpoints) or degree-three nodes (bifurcations), while the path between the segment endpoints consisted of degree-two nodes (centerlines). The ends of the segments were added as nodes in G_d while the remaining (interior) points on the segment were added as an attribute of the edge connecting the two nodes. This algorithm is described in detail in Fig. 7.

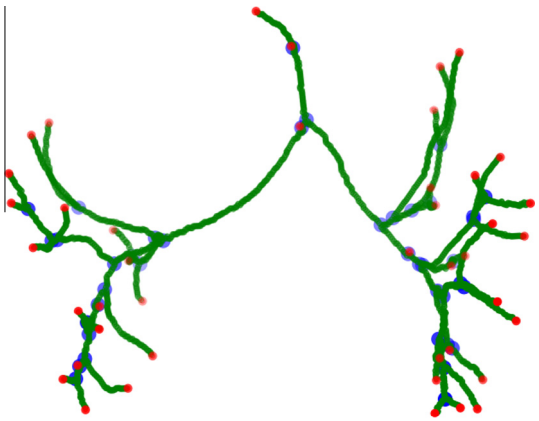


Fig. 6. Example undirected graph. Degree-one nodes (endpoints) are drawn in red, degree-two nodes (centerlines) are drawn in green, and degree-three nodes (bifurcations) are drawn in blue. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

```
input  : G_u
output: G_d
n_r = FindRMSRoot (G_u);
S_nb = FindBifurcations (G_u);
S_ne = FindEndpoints (G_u);
foreach n_e ∈ S_ne do
    // Find shortest path
    P_s = BidirectionalDijkstra (G_u, n_r, n_e) ;
    n_s = n_e;
    e = [];
    foreach n_p in P_s do
        if n_p ∈ S_nB then
            // split the path; add nodes with edge to G_d
            AddDirectedEdge (G_d, n_p, n_s, e);
            n_s = n_p;
            e = [];
        end
        else
            e = e.append(n_p)
        end
    end
end
```

Fig. 7. Algorithm for generating directed graph G_d from an undirected graph G_u.

A directed graph generated from the undirected graph in Fig. 6 is shown in Fig. 8.

2.3.3. Voxel mapping

Once G_d was generated, we mapped each voxel from the original segmentation to the nearest edge $e_{i,j}$ in the directed graph. Voxel-to-edge mapping used local coordinate systems defined along each edge centerline. The local coordinate systems were based on cubic least-squares spline fits to the edges of the centerlines (inclusive with the nodes connected by each edge). The spline was sampled at points x_k where the first derivative (D_k^1) of the splines represented the tangent of the curve at each sampled point and thus the local direction of the centerline. Corresponding orthogonal planes were defined using the Hessian normal form of a plane. Let $\vec{n_k}$ be the unit vector parallel to D_k^1 and let $\vec{x_k}$ be

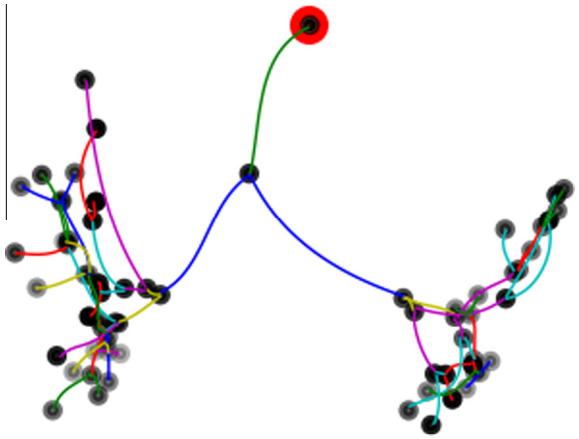


Fig. 8. Example directed graph. In this graph, the root node (n_r) is outlined in red. All other nodes are drawn in black. The color of each edge (centerline) indicates the depth (as measured from the terminating node of the edge) from the root node. Green denotes a depth of 1, blue a depth of 2, etc. Seven colors were used. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

the vector from the origin to the point x_k, then the plane orthogonal to $\vec{n_k}$ at x_k is defined by the residual value p_k as follows:

$$\vec{n_k} \cdot \vec{x_k} = -p_k. \quad (2)$$

All points on the plane share the same residual value (p_k). Thus along every sampled point on the fitted centerline we recorded $\vec{n_k}$ and p_k. For any voxel at world coordinate x, y, z ($v(x, y, z)$) in the segmented image, we map the voxel to the centerline point x_k such that

$$\forall v \in S_{\text{seg}} : \underset{k}{\text{minimize}} \, \|p_k - (\vec{n_k} \cdot \vec{v})\|. \quad (3)$$

To summarize, G_d is a directed graph representing the vascular structure we segmented. A node n_i in G_d represents either a bifurcation or endpoint in the segmentation. An edge $e_{i,j}$ in G_d represents the centerline connecting nodes n_i and n_j. At each point x_k along $e_{i,j}$ we record the local direction of the centerline $\vec{n_k}$, the residual p_k defining the orthogonal plane, and the set of all voxels in the original segmentation that lie in that plane.

An example of a final Graph (G_d) is shown in Fig. 11.

2.3.4. Graph pruning

Unfortunately, even with preprocessing prior to 3D parallel thinning, the graph generation process described above can still produce graphs with a number of errors, most notably false centerline segments due to imperfections in the surfaces of the segmentations. In previous work on automated graph generation in intracranial vessels [5], we found that automatically pruning centerlines shorter than five voxels was an effective heuristic. With the pulmonary vasculature, automated pruning is particularly challenging because of the wide range of vascular diameters present and the short length (relative to the diameter) of the segments in the tree, resulting in actual centerlines being deleted when the threshold is made large enough to eliminate false centerlines in the pulmonary trunk. (See Fig. 9.)

We therefore explored whether pruning could be determined with a machine learning approach. To create a data set for machine learning, we visually reviewed each of the directed graphs G_d in our training and test sets, manually deleting any degree-one nodes (with the accompanying edge) that were judged to be extraneous. The review was done using a Mayavi [21] script based on VTK [12] that rendered the graph nodes, the fitted centerlines, and the surface of the mapped voxels. All voxels that had been mapped to a deleted edge were remapped to existing edges and the review continued until no more edges remained to be deleted. Based on this editing, every edge existing in our collection of ordered graphs was labeled as deleted or not deleted. These labels then became the target of the machine learning classifiers. (See Fig. 10.)

We used the open source, Python machine learning software package scikit-learn [22] to generate both random forest [23] and logistic regression classifiers for identifying the edges that should remain or be deleted from the graphs. For the classification task, we extracted edge features from the un-pruned graph that we believed would differentiate true vascular segments from false vascular segments. The features were extracted from the centerlines themselves (e.g. path length) and from the voxels mapped to the edges (e.g., surface2volume). These features, their explanations, and brief justifications, are shown in Table 1.

In addition to these features, we also created interaction terms between depth and each feature, since vascular characteristics vary with depth in the pulmonary tree.

The classifiers were trained by optimizing the area under the ROC curve (AUC). We used recursive feature elimination with cross validation [24], as implemented in the scikit-learn function *REFC*, to select the optimal feature set. However, we found that the *REFC* performance was highly dependent on the characteristics of the classifier (e.g. the number of estimators in the random forest) as well as the number of cross validations performed. We therefore examined feature selection using a repeated number of REFC runs with varying the number of cross validations (2, 4, 6, 8, 10) and varying number of estimators in the random forest (10, 20, 40, 80, 160, 320, 640, 1280). The fraction of runs where a given feature was selected for the classifier was the figure-of-merit. We built multiple models where the threshold for inclusion was varied from zero (all features were included in the model) to 1 (no features were included in the model). We then selected the most parsimonious model (fewest parameters achieving highest AUC).

2.4. Semantic labeling

We provided semantic labeling of the pulmonary trunk and left and right pulmonary arteries in our graph. Our semantic labeling proceeds from the graph root identification described in Section 2.3.2. Given the root of the graph (n_r), which we assume represents the superior portion of the pulmonary trunk (n_{PT}), we label the edge between the root and its child node (n_1) as the pulmonary trunk. The children nodes of n_1 are then examined. The edge between n_1 and its right-most child node (n_r) is labeled as the right pulmonary artery and the edge between n_1 and its left-most child nodes (n_l) is labeled as the left pulmonary artery.

2.5. Evaluation

Using the best performing machine learning classifier, we automatically pruned the vascular graphs. The machine learning classifier was trained using a leave-one-out approach. That is, the classifier was trained with data from all the cases except the case to be pruned. Using this automatically pruned graph, we then identified the pulmonary trunk and left and right pulmonary arteries.

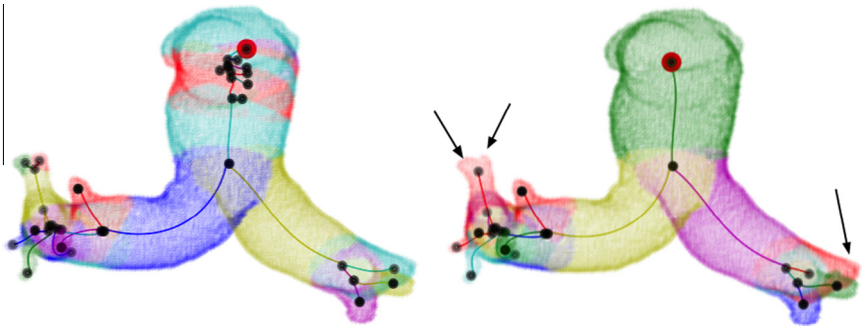


Fig. 9. Example graphs showing limitations of length-based pruning. On the left is the graph generated with default pruning length of 5 voxels. On the right is the graph generated by increasing the pruning length to 20 voxels in order to eliminate the false centerlines in the pulmonary trunk. The resulting graph has lost true segments (marked by arrows) due to this more aggressive pruning.

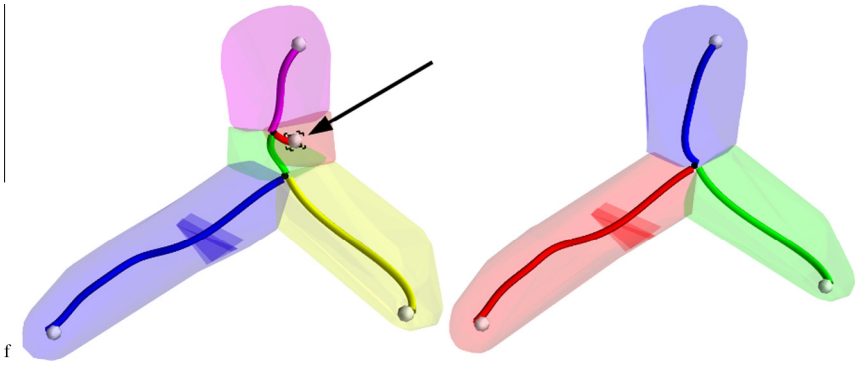


Fig. 10. Illustration of manual pruning for creating data set. A spurious centerline in the pulmonary trunk is selected (arrow) and deleted. The voxels mapped to that centerline are remapped and a single centerline for the pulmonary trunk is produced.

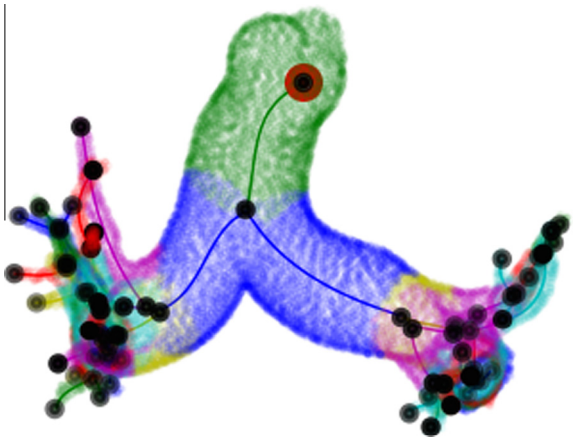


Fig. 11. Rendering of an example graph. This was the largest graph generated in our data set. The root node is outlined in red; all other nodes are shown in black. The voxels from the original segmentation have been mapped to particular edge centerlines. Centerlines and matching surfaces have been drawn with matching colors corresponding to the depth of the edge. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

Table 1
Vascular features for pruning.

Feature	Explanation	Justification
path length	Number of voxels in edge (centerline)	False centerlines tend to be shorter than the vessels themselves
exterior2surface	Ratio of exterior surface of segment to total surface of segment	Measures how a false segment "drills into" a true segment
surface2volume	Ratio of total surface of segment to the volume of the segment	Measures deviation from cylindricalness
mindfe	Minimum distance-from-edge along centerline	Measures proximity to the surface
maxdfe	Maximum distance-from-edge along centerline	Measures proximity to the surface
minmaxdfe	mxdf–mindf	Measures constancy of vessel diameter
lngth2wdth	Ratio of the length of the centerline to the average width of the segment	Measures deviation from cylindricalness
angle	The branching angle (in radians) between the centerline and its parent edge	Measures deviation from physiological branching angles
depth	The length (in millimeters) of the minimum path from the terminating node of the centerline to the graph root	Generated for use as an interaction term to account for feature variation by tree depth

The accuracy of this semantic labeling was compared to that obtained with the graphs prior to the automatic pruning. Differences in the paired proportions were then tested. We also compared correlation between manual and automatically measured diameters of the three labeled pulmonary arteries using both the non-pruned and pruned graphs.

2.5.1. Semantic labeling of vascular anatomy

We reviewed whether vascular anatomy could be labeled directly from the unpruned graphs or if pruning was first necessary to identify each part. Review was done by generating a 2D projection of G_d with the nodes, edges (least-squares fitted centerlines), and the mapped surfaces. From these 2D projective images we then assessed whether (1) n_{PT} was correctly identified, (2) whether n_1 was identified correctly, and (3) whether n_L and n_R were identified correctly. For n_R and n_L, correct identification meant that the nodes were placed near to the origin of the right and left apical and anterior segmental arteries.

2.5.2. Correlation with manual diameter measurements

Manual artery diameter measurements (pulmonary trunk and left and right pulmonary artery) were made by four reviewers. One of the authors manually selected a slice from each subject for measurement. These slices were presented to four reviewers in OsiriX (http://www.osirix-viewer.com/) where a line region-of-interest tool was used to measure the diameter of each vessel at the perceived position of maximum width. Each observer was blind to both the other reviewers' results and the quantification from the vascular models. The average of the reviewers' measurements for each case was used for comparison to the automated measurements.

Automated artery diameter measurements were made by identifying the midpoint of each segment with its corresponding orthogonal plane (see Section 2.3.3). The average distance of surface points mapped to this plane was taken to be the radius of the segment. Differences in the paired correlation coefficients between the average manual diameter and the automated measurements were then tested with Williams' test, a test for comparing correlated correlation coefficients [25].

2.6. Computational environment and profiling

All Python code was run using the Anaconda (Continuum Analytics) Python 2.7.8 release. Relevant packages included Pandas (version 0.14.1), scikit-learn (version 0.14.1), IPython

(version 2.3.1), SimpleITK (version 0.8.0), NumPy (version 1.9.1), and Insight Toolkit (version 3.2.0).

Graph generation was performed on an Ubuntu 14.04 workstation with 32 GBytes of RAM and 16 Intel Xeon CPUs (E5-2620 v2 @ 2.10 GHz). Computational performance of the algorithms were assessed on 25 randomly selected cases. The line_profiling Python package was used to measure total computation time for each algorithm function, and the percent of time spent on each line of the functions.

3. Results

We first review the evaluation of preprocessing of the segmentation. Then we review the accuracy of anatomy identification on our unpruned vascular graphs. Finally we describe the performance of machine learning for identifying extraneous edges.

3.1. Segmentation preprocessing

Preprocessing was found to be essential, reducing the number of edges per graph from approximately 52 to roughly 18. The best performance was achieved with the combination of median filtering and morphological closing with a larger kernel size. Table 2 summarizes these preprocessing results.

3.2. Machine learning to predict graph pruning

3.2.1. Feature selection

We ran the feature selection for random forests 40 times and looked at the fraction of times each feature was selected. The results are shown in Table 3.

A similar approach for selecting features for logistic regression, although without iterating over the number of classifiers, was used. Results for the logistic regression feature selection are shown in Table 4.

Grouping the features together by the proportion of times they were selected for the model (e.g. 0.775), we built random forest classifiers and evaluated them using cross validation. For both logistic regression and random forests, the performance of the classifiers did not drop as we removed features up to the group of features that were always selected.

Both the random forest and logistic regression had relatively high ROC areas (approximately 0.98). To evaluate if there were statistical differences between the classifiers, we used McNemar's test. Using the smallest feature set for each classifier, we compared the performance of the classifiers with training/testing test sizes of (100, 653), (200, 553), (300, 453), (400, 353), and (500, 253). There were no statistically significant differences between the classifiers. Since, the random forest classifier used a smaller feature set, we selected it as our classifier.

We examined whether a smaller set of features could be used with the random forest classifier. We were particularly interested in whether the pruning could accurately be done with only centerline/edge features as this would reduce the computational burden of mapping and remapping surface and volume points to the graph model. We repeated our feature selection process described above using only the top six selected features. The selection proportions are shown in Table 5. We built three different random forest classifiers: (1) using the three features (T3) that were always selected ("pathlength:depth","exterior2surface:depth","length2width:depth"), (2) using edge-only (EO) features ("pathlength:depth","pathlength","depth:depth"), and (3) using only "pathlength" (PO). The area under the ROC curves (AUC) are shown in Table 6. As can be

Table 2
Preprocessing of the segmentation data.

	Avg. no. of edges per case
No preprocessing	52.08
Median only	18.32
Close only	18.14
Median and close (small kernel)	20.46
Median and close (large kernel) (2, 2, 2)	16.92

Table 3
Feature selection for the random forest classifier.

Variable	Proportion selected
pathlength	1.000
exterior2surface	1.000
pathlength:depth	1.000
exterior2surface:depth	1.000
length2width:depth	1.000
depth:depth	1.000
depth	0.975
surface2volume	0.850
length2width	0.775
maxdfe:depth	0.775
maxdfe	0.575
angle:depth	0.575
mindfe:depth	0.450
minmaxdfe:depth	0.425
surface2volume:depth	0.350
mindfe	0.300
angle	0.100
minmaxdfe	0.075

Table 4
Feature selection for logistic regression.

Variable	Proportion selected
pathlength	1.0
exterior2surface	1.0
surface2volume	1.0
mindfe	1.0
maxdfe	1.0
minmaxdfe	1.0
length2width	1.0
depth	1.0
pathlength:depth	1.0
exterior2surface:depth	1.0
surface2volume:depth	1.0
mindfe:depth	1.0
length2width:depth	1.0
angle:depth	1.0
maxdfe:depth	0.8
minmaxdfe:depth	0.5
angle	0.2
depth:depth	0.0

Table 5
Feature sub-selection for top six features.

Variable	Proportion selected
pathlength:depth	1.000
exterior2surface:depth	1.000
length2width:depth	1.000
pathlength	0.950
depth:depth	0.850
exterior2surface	0.575

Table 6
ROC areas for feature subsets from top six features.

Variable group	AUC
Top-three features	0.975408
Edge-only features	0.957648
Pathlength only	0.829489

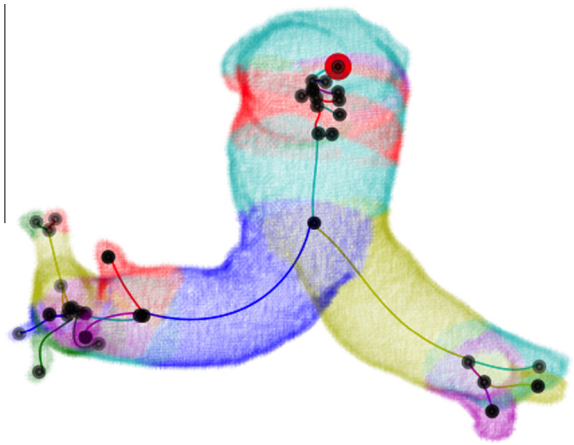


Fig. 12. Rendering of a graph where the identified pulmonary trunk root node was suboptimal.

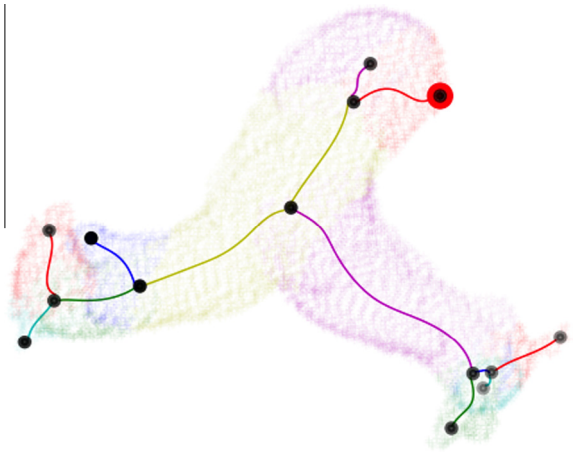


Fig. 13. Rendering of a graph where the identification of n_1 (and consequently n_R and n_L) is due to false centerlines in the pulmonary trunk.

seen, pathlength only shows a significant drop in performance. However, using only the top three features or using edge only features still obtains high classification performance.

3.3. Anatomy identification

3.3.1. Root identification

n_{PT} was correctly identified in 114 out of 116 cases. For the two erroneous cases, the identified node was adequate but a nearby node would have been preferred. An illustration of this is shown in Fig. 12. n_1 was correctly identified in the non-pruned graphs in 82 out of 116 cases, failures being due to the presence of false centerlines between n_{PT} and the true n_1 location, as illustrated in Fig. 13. n_R and n_L were both correctly identified in 74 out of the 116 cases respectively. The majority of these failures were due to a failure to first correctly identify n_1.

Table 8
Statistical analysis of semantic labeling for top-performing algorithms.

		Top-six vs top-three	Edge-only vs top-three	Edge-only vs top-six
Main	p-value	0.317	0.564	0.0833
	z	1	−0.577	−1.73
Left	p-value	0.317	0.655	0.180
	z	1	−0.447	−1.34
Right	p-value	0.317	0.564	0.564
	z	1	0.577	−0.577

3.3.2. Semantic labeling of vascular anatomy with and without pruning

Performance of the semantic labeling of the vascular anatomy were similar for all three arteries examined. Results are shown in Table 7. Semantic labeling with pruning performed better than with the unpruned graph regardless of the pruning algorithm. These differences were highly statistically significant ($p \ll 0.05$) with correction for multiple comparisons. Pathlength-only pruning performed significantly worse the other pruning algorithms, as expected. There was not a statistically significant difference between the edge-only (EO) and top-three-feature (T3), and all six top features (T6) pruning. Statistical comparisons of the top three performing pruning algorithms are shown in Table 8. There were no significance differences between the three algorithms.

3.4. Morphological feature extraction using automatically generated graphs

Automated pruning of the graphs uniformly improved correlation coefficients between manual and automated measurements. Table 9 shows the correlation coefficients between the manual and automated (with and without pruning) measurements for the three arteries of interest. Correcting for multiple comparisons, the improvements in the pulmonary trunk correlation were nearly statistically significant ($t = -2.047$, $p = 0.022$, $\alpha = 0.017$) while the improvements for the left and right pulmonary arteries were highly significant ($p \ll 0.017$).

Table 7
Accuracy of semantic labeling of pulmonary arteries from graphs.

		No pruning	Edge-only	Pathlength-only	Top-six	Top-three
Main	Correct	82	112	91	113	115
	Wrong	34	4	25	3	1
	Prop. correct	0.707	0.965	0.784	0.974	0.991
	Total	116	116	116	116	116
Left	Correct	69	111	84	112	114
	Wrong	47	5	32	4	2
	Prop. correct	0.595	0.957	0.724	0.966	0.983
	Total	116	116	116	116	116
Right	Correct	73	111	89	110	112
	Wrong	43	5	27	6	4
	Prop. correct	0.629	0.957	0.767	0.948	0.966
	Total	116	116	116	116	116

Table 9
Correlation coefficients for manual and automated diameter measurements.

	Pruned	Original
Main manual	0.814	0.762
Left manual	0.769	0.592
Right manual	0.790	0.579

Table 10
Average computation time (in seconds) over 25 cases.

Skeleton generation	Graph generation	Voxel mapping
107.689	2.046	28.344

3.5. Computational performance

We performed code profiling on 25 randomly selected cases. The average compute times for the various steps are shown in Table 10. These values were computed without parallel processing. Many of the steps are obviously parallelizable and thus the algorithm could be expected to have increased performance when this is exploited.

In the graph generation step the two most expensive steps were (1) labeling of the binary image to get the connected components (55.5%) and (2) generating the undirected graph from the labeled skeleton image (33.3%). In the voxel mapping, the most computationally expensive step that we have created, the two most expensive steps were (1) mapping the surface voxels to the edges (72.0%) and binning the mapped voxels into orthogonal planes (21.6%).

4. Discussion

In this paper we have addressed the segmentation, structuring, and labeling of the central pulmonary arteries depicted in CTPA images using vascular graphs. While we limited ourselves to CTPA images, the techniques would be applicable to magnetic resonance images. Our work was motivated by the goal of automated characterization of pulmonary hypertension by quantifying the diameters of the pulmonary trunk and the left and right pulmonary arteries, but the techniques we use are obviously applicable to other vascular trees. In fact, our work here expands upon similar work we have done with intracranial vessels. Application to many other vascular trees can be envisioned. For example, capturing the hepatic arteries and portal veins for planning liver resections or analyzing the deep veins of the pelvis and lower extremities for identifying thrombosis. An obvious extension of our work is to the structuring of the pulmonary airways, which share geometric similarities to the pulmonary arteries. In our limited experience with airway segmentations, there are a number of notable differences in what challenges need to be addressed, primarily that while the contrast between the HU of air and surrounding tissues is high with little or no overlap, airway segmentation is easily disrupted by artifacts. Simultaneously segmenting both the pulmonary arteries and the pulmonary airways would help us with our pruning and semantic labeling as the arteries and airways are bundled with roughly equal diameter. The pairing of arteries and airways could thus help distinguish artery from vein and help eliminate false positives by having reinforcing information about size and direction.

In this work we used the existing ITK-SNAP package to perform a semi-automated segmentation process, as described in Section 2.2. Even with great care in the selection of segmentation parameters, it was extremely difficult to consistently segment the pulmonary arteries while excluding with high specificity the aorta, heart, veins, and other structures. Consequently, we manually edited the segmentations to eliminate spurious components of the vascular segmentation. Our starting segmentations will therefore likely be of higher quality than what would realistically be expected from a fully automated method. Nonetheless, our segmentations still retain imperfections, and an important question we address is how to achieve accurate semantic labeling of the central pulmonary arteries despite these imperfections. Since we had corrected gross segmentation errors (primarily bleeding of the pulmonary artery segmentation into the aorta) by hand, we address how to work with only a modest amount of local imperfections in the segmentation. Further, we are only concerned with how these imperfections impact our skeleton generation and subsequent semantic labeling.

The extent that these errors occur is tightly related to the type of segmentation used, specifically in how much local, neighborhood, or global information is incorporated into the segmentation. The geometric level set segmentation method we used requires the user to balance how much neighborhood information (smoothing) is enforced on the segmentation. In order to promote the segmentation proceeding down the distal arteries, we opted for a smaller constraint on the surface smoothness. Consequently, our pulmonary artery segmentation was more prone to leak out into adjacent structures. Since our focus was on the central pulmonary arteries, we likely could have increased the amount of smoothness enforced on the segmentation, thus reducing the number of false segments generated, and still segmented the arteries we analyzed. However, since distal vessels are ultimately also of interest, a more elegant solution would be to incorporate global information *a prior* in the form of statistical shape models of the pulmonary arteries.

For our skeleton generation, we also used an existing parallel thinning package. The parallel thinning package is freely available and ran quickly, but may be more sensitive to surface irregularities than techniques that use more global information. Our pre-processing steps reduced but did not completely eliminate errors in the graph generation due to surface irregularities, primarily due to the fact that we are still working with the discrete nature of the images. An alternative technique worth exploring would be surface smoothing techniques such as those available in SPHARM-PMD [26]. We did show, however, that machine learning techniques can be employed to accurately prune false edges from the graphs with ROC areas as high as 0.98. Using the machine learning algorithms to prune the graphs, we were able to achieve accurate semantic labeling of the main pulmonary arteries. The main pulmonary trunk was correctly identified as high as 99% of the time with the left and right pulmonary arteries identified correctly 98% and 97% of the time respectively. The most accurate pruning relied on features from both volumetric, surface, and centerline edge features, but accurate pruning can still be achieved using edge-only features which are much less costly to compute. The trade-off between accuracy and computational cost is an open question that will be application dependent. In our classification models, interaction terms with depth (path distance from the graph root to the edge) were vital. Adding depth and depth interaction terms significantly increased the AUC of our pruning models compared to earlier results [27] where our highest ROC area was only 0.89. To our surprise branching angle, which had been suggested as a valuable feature to differentiate true from false centerlines [6], was not an informative feature.

When building our pruning classifiers, we did not observe great performance with feature selection algorithms, as described in Section 3.2.1. This might be due to the fact that our feature space was already fairly small and the algorithms were designed for reducing much larger feature sets. The random nature of the feature selection process implies variability in the resulting selected feature sets. We have observed this phenomena but have not systematically investigated it.

There are several limitations of this work. Foremost, while our intent is to develop an automated method, we are still limited to a fairly manual segmentation process. While we are working on developing automated segmentation techniques for the pulmonary arteries, a viable solution remains elusive. We do not intend for our analysis to always rely on hand edited segmentation, but believe our results will be similar to those obtained using more sophisticated techniques that may be developed in the future. The vascular segmentation and editing of the segmentation were all done by a single user. Thus our models are more susceptible to human error than if we had used the consensus results of multiple observers. Related to this, our visual review of the models for pruning only presented the model and was not integrated with the original 3D images. Thus it could at times be difficult to differentiate spurious centerlines from true vascular stubs. Given the wide anatomic variation in the pulmonary arteries, our data set is relatively small. Further, we only assessed anatomic identification and morphology measurements for the central pulmonary arteries.

The work presented here points towards several obvious next technical steps. First, we should investigate whether the machine learning pruning could be used to prune vascular skeletons without manual deleting of the regions where the segmentation bleeds into non-arterial regions. If this approach were successful, a greater latitude on segmentation accuracy could be tolerated. Second, the graphs themselves could form the basis for a segmentation method where the graphs represent *a priori* information about the pulmonary vascular anatomy. Finally, we should investigate the characterization of a broader range of features from the pulmonary arteries, and explore the use of these features for disease diagnosis and phenotyping.

5. Conclusion

We have presented a graph-based technique for automatically identifying vascular substructures within a segmentation of a vascular tree. We illustrated our technique by applying it to pulmonary arterial trees. Our technique leverages a variety of open source, platform independent tools that greatly reduced the cost of development and improved the quality of the resulting code, since we were able to leverage well written, validated code bases such as the Insight Toolkit. Our project is available on GitHub at https://github.com/chapmanbe/vasctree.

In work to be reported separately, we have used these automatically generated models to both classify subjects as being positive or negative for pulmonary hypertension as well as to predict pulmonary arterial pressures. We are currently working on using these graphs as a data-reduction method for characterizing large populations of vascular trees, with application to image retrieval, knowledge-based skeletal pruning, and computer-assisted diagnosis.

References

[1] S.R. Aylward, E. Bullitt, Initialization, noise, singularities, and scale in height ridge traversal for tubular object centerline extraction, IEEE Trans. Med. Imaging 21 (2) (2002) 61–75.

[2] J.P. Janssen, G. Koning, P.J. de Koning, J.G. Bosch, J.C. Tuinenburg, J.H. Reiber, A new approach to contour detection in X-ray arteriograms: the wavecontour, Invest. Radiol. 40 (8) (2005) 514–520.

[3] I. Bitter, A.E. Kaufman, M. Sato, Penalized-distance volumetric skeleton algorithm, IEEE Trans. Visual. Comput. Graphics 7 (3) (2001) 195–206, http://dx.doi.org/10.1109/2945.942688. http://dx.doi.org/10.1109/2945.942688.

[4] C.M. Ma, M. Sonka, A fully parallel 3d thinning algorithm and its applications, Comput. Vis. Image Underst. 64 (3) (1996) 420–433, http://dx.doi.org/10.1006/cviu.1996.0069. http://dx.doi.org/10.1006/cviu.1996.0069.

[5] L. Zhang, B.E. Chapman, D.L. Parker, J.A. Roberts, J. Guo, P. Vemuri, S.M. Moon, F. Noo, Automatic detection of three-dimensional vascular tree centerlines and bifurcations in high-resolution magnetic resonance angiography, Invest. Radiol. 40 (10) (2005) 661–671.

[6] A. Szymczak, A. Stillman, A. Tannenbaum, K. Mischaikow, Coronary vessel trees from 3D imagery: a topological approach, Med. Image Anal. 10 (4) (2006) 548–559.

[7] A. Devaraj, A.U. Wells, M.G. Meister, T.J. Corte, S.J. Wort, D.M. Hansell, Detection of pulmonary hypertension with multidetector CT and echocardiography alone and in combination, Radiology 254 (2) (2010) 609–616.

[8] A. Grubstein, O. Benjaminov, D.B. Dayan, D. Shitrit, M. Cohen, M.R. Kramer, Computed tomography angiography in pulmonary hypertension, Isr. Med. Assoc. J. 10 (2) (2008) 117–120.

[9] P.D. Edwards, R.K. Bull, R. Coulden, CT measurement of main pulmonary artery diameter, Br. J. Radiol. 71 (850) (1998) 1018–1020.

[10] S. Rajaram, A.J. Swift, R. Condliffe, C. Johns, C.A. Elliot, C. Hill, C. Davies, J. Hurdman, I. Sabroe, J.M. Wild, D.G. Kiely, CT features of pulmonary arterial hypertension and its major subtypes: a systematic CT evaluation of 292 patients from the ASPIRE registry, Thorax (2014).

[11] T.S. Yoo, Insight into Images: Principles and Practice for Segmentation, Registration, and Image Analysis, AK Peters Ltd, 2004. ISBN: 1568812175.

[12] W. Schroeder, K. Martin, B. Lorensen, The Visualization Toolkit, third ed., Kitware Inc., 2004. ISBN: 1930934122.

[13] J.E. Cates, R.T. Whitaker, G.M. Jones, Case study: an evaluation of user-assisted hierarchical watershed segmentation, Med. Image Anal. 9 (6) (2005) 566–578.

[14] H.C. Shin, M.R. Orton, D.J. Collins, S.J. Doran, M.O. Leach, Stacked autoencoders for unsupervised feature learning and multiple organ detection in a pilot study using 4d patient data, IEEE Trans. Pattern Anal. Machine Intell. 35 (8) (2013) 1930–1943. doi:http://dx.doi.org/10.1109/TPAMI.2012.277.

[15] R. Donner, B.H. Menze, H. Bischof, G. Langs, Global localization of 3D anatomical structures by pre-filtered Hough forests and discrete optimization, Med. Image Anal. 17 (8) (2013) 1304–1314.

[16] P.A. Yushkevich, J. Piven, H. Cody Hazlett, R. Gimpel Smith, S. Ho, J.C. Gee, G. Gerig, User-guided 3D active contour segmentation of anatomical structures: significantly improved efficiency and reliability, Neuroimage 31 (3) (2006) 1116–1128.

[17] R.C. Gonzalez, R.E. Woods, Digital Image Processing, second ed., Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 2001. ISBN: 0201180758.

[18] E.R. Dougherty, R.A. Lotufo, Hands-on Morphological Image Processing (SPIE Tutorial Texts in Optical Engineering, vol. TT59), SPIE Publications, 2003. ISBN: 081944720X.

[19] H. Homann, Implementation of a 3d thinning algorithm, Insight J. (2007).

[20] A.A. Hagberg, D.A. Schult, P.J. Swart, Exploring network structure, dynamics, and function using networkx, in: G. Varoquaux, T. Vaught, J. Millman (Eds.), Proceedings of the 7th Python in Science Conference, Pasadena, CA, USA, 2008, pp. 11–15.

[21] P. Ramachandran, G. Varoquaux, Mayavi: 3D visualization of scientific data, Comput. Sci. Eng. 13 (2) (2011) 40–51. doi:http://dx.doi.org/10.1109/MCSE.2011.35.

[22] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, E. Duchesnay, Scikit-learn: machine learning in python, J. Machine Learn. Res. 12 (2011) 2825–2830.

[23] L. Breiman, Random forests, Machine Learn. 45 (1) (2001) 5–32, http://dx.doi.org/10.1023/A:1010933404324. http://dx.doi.org/10.1023/A%3A1010933404324.

[24] I. Guyon, J. Weston, S. Barnhill, V. Vapnik, Gene selection for cancer classification using support vector machines, Machine Learn. 46 (1–3) (2002), http://dx.doi.org/10.1023/A:1012487302797. http://dx.doi.org/10.1023/A:1012487302797.

[25] D. Howell, Statistical Methods for Psychology, Thomson Wadsworth, 2007. ISBN: 9780495093619. <https://books.google.com/books?id=-bmMPwAACAAJ>.

[26] M. Styner, I. Oguz, S. Xu, C. Brechbuehler, D. Pantazis, J. Levitt, M. Shenton, G. Gerig, Framework for the statistical shape analysis of brain structures using spharm-pdm, Insight J. (2006).

[27] H. Berty, Semi-automated diagnosis of pulmonary hypertension using puma, a pulmonary mapping and analysis tool, 2013. <http://d-scholarship.pitt.edu/18190/>.